IDACT: Automating Data Discovery and Compilation

Kara L. Nance and Brian Hay
Department of Mathematical Sciences
University of Alaska Fairbanks
Box 756660
Fairbanks, AK 99775-6660

Abstract - **IDACT serves as an automated middle-layer which acts as an interface between the user and the heterogeneous data sources. Data sources are registered with the middle-layer, and a user can then request data from any (or all) of the available sources without having knowledge of the specific implementation details and formats. The data returned will be automatically converted from the native format to the formats required by the scientific models or analysis tools. Finally, the intermediate data and the products of the models may be archived in a data warehouse for re-use in the future. This approach allows the researcher to focus on the analysis of results, while ensuring that both the hardware and the researcher's time are used as efficiently as possible.**

I. Introduction

As scientific models and analysis tools become increasingly complex they allow researchers to manipulate larger, richer, and more finely-grained datasets, often gathered from diverse sources. These complex models provide scientists with the opportunity to investigate phenomena in much greater depth, but this additional power is not without cost. Often this cost is expressed in the time required on the part of the researcher to find, gather, and transform the data necessary to satisfy the appetites of their data-hungry computational tools, and this is time that could clearly be better spent analyzing results. In many cases, even determining if a data source holds any relevant data requires a time-consuming manual search, followed by a conversion process in order to view the data. Frequently as the researcher spends time searching for and reformatting the data, the expensive hardware remains underutilized. In addition, the heterogeneous, distributive, and voluminous nature of many government and corporate data sources imposes severe constraints on meeting the diverse requirements of users who analyze the data [4]. Fortunately it is possible to automate much of this process, allowing for increased productivity on the part of the scientists, and increased throughput and utilization for the models and analysis tools. This research effort builds on proven technology in automating the compilation of heterogeneous scientific datasets [4, 5, 6, 7, 8, 9, 10, 11] to create an Intelligent Dataset Identification, Assimilation, Collection and Transformation Systems (IDACT).

II. System process overview

A high level description of the IDACT solution involves eight steps, which are depicted in figure 1. To start the process (1) the researcher submits a standard query to the IDACT layer. This query can be submitted using a variety of formats designed to meet the needs of the particular user, ranging from a basic pick-list type interface to a highly detailed SQL type statement. At this point (2), the IDACT system will use the information in the Data Source Registry to automatically generate a series of queries that will allow the data sources to be identified and searched. In some cases, these queries will be SQL statements for access to a relational database system, although in other cases the "queries" may be as simple as checking a directory structure for files. In step (3), the queries generated by IDACT will executed in order to retrieve (4) any relevant data. It should be noted that the system does not only query the external data sources, but also the internal data warehouse that will be used to store commonly used transformed data, model outputs, and analysis tool results.

The data returned to IDACT (4) will meet one of three conditions, which will be handled by the Transformation Manager. The easiest case is the one in which the data is already in the format required by the user. Slightly more complex to deal with is data that is not in the required format, but for which a previously defined conversion mechanism is available. Several of these commonly used conversion mechanisms already exist, such as from *netCDF* to *HDF5*, and others will be developed as part of this project. The third possibility involves the case in which the data requires a conversion, but no defined conversion exists. This is the most complex case, but it is simplified considerably from the perspective of the user through the use of automated intelligent agents to develop a proposed conversion process, which is then presented to the user for either approval or modification. Once a conversion process is identified, the data can clearly be converted to the required format. However, a major strength of the IDACT system lies in its ability to "save" this conversion process for automatic use with similar datasets in the future.

When all of the necessary data components are available in a suitable format (5), the Scheduler determines when each model and analysis tool should be run to achieve maximum throughput while ensuring that each meets its scheduled completion deadline (6). This step may be omitted in the event that the researcher wants to use IDACT to simply gather and transform data, which will be used as input to an offline model or analysis tool. Although a factor in the

This research is funded through the NASA Advanced Information Systems Technology Program (NASA award AIST-02-0135)

scheduler will be the completion deadline for each model run, it will not be known in advance exactly how long a model will take to execute. This problem can be solved using a learning algorithm. By comparing the characteristics of previous models runs with their execution time, it will be possible to estimate the time required to complete the current models, and this estimate will be used in the scheduling decisions.

The output from the model or tools is then returned to the researcher (7), allowing them to analyze the results. Often these results will be archived (8) in the IDACT data warehouse for use in future research efforts.

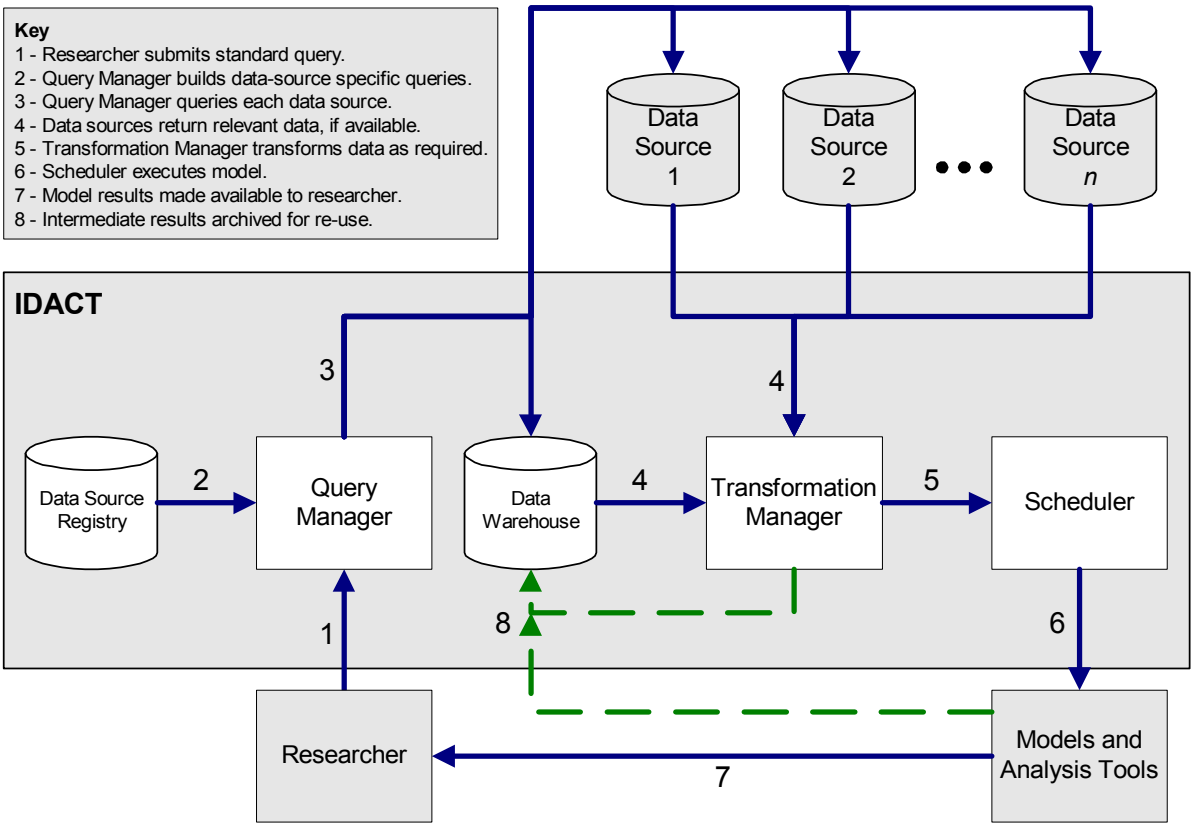

Figure 1 - High Level System Design

III. Data Discovery

The Data Source Registry (DSR) is the component of the system that facilitates the data discovery process targeting domain-specific applications [1]. The DSR allows the identification of domain-specific datasets as candidates for combination, as well as the identification of relationships between datasets. Version 1 of the DSR uses a data pull method with the essential metadata being stored in the data source registry. It allows creation of a set of common fields, to which fields from individual datasets can be mapped using the various classes of transformations identified to support the DTM. A community of users can add identified datasets to the DSR to provide a centralized data portal. Version 2 of the system will provide mechanisms for automating the data registry candidate identification step to further strengthen the domain datasets available for data mapping.

IV. Data Compilation

The Data Transformation Manager (DTM) component is the center of the data compilation process [3]. The DTM accepts the results of a data query, and then determines and applies the appropriate conversion to apply in order to obtain the required output format. Consider the example in which the DTM receives an XML document as a result of a query. In such a case, the DTM may face one of the following possible situations, which are presented in order of increasing complexity:

A. The easiest situation is the one in which the input and output format are the same, and as a result the DTM simply passes the input data directly to the output location.

B. A slightly more complex situation is the one in which the input and output formats differ, but where the DTM is aware of a transformation process that can be applied to convert between the two formats. The DTM searches its internal database to determine if a suitable transformation exists, and applies the transformation if one is found. For example, the input document may be valid with respect to one XML Schema (*.xsd* file), whereas the required format is an XML document that is valid with respect to a different XML Schema. If an XSLT document which performs the required transformation is available to the DTM, then it can be applied to the input document via an XSLT processor, such as Apache *Xalan*, resulting in the required output format. Transformations are not limited to XSLT documents, however, and the DTM currently allows other transformation methods, such as executables, scripts, and database updates, to be used.

C. If a single transformation cannot be found by the DTM, it next attempts to build a transformation sequence which results in the required output format. For example, if the DTM is attempting to convert format *A* into format *C*, it may not be aware of a method that directly performs the transformation. However, if the DTM is able to transform format *A* into format *B* using a known method, and is also able to transform format *B* into format *C* using another known method, and the required transformation can be achieved by applying these two transformations in sequence. In this manner, the DTM can perform transformations for which a single transformation process has not been defined.

D. If a suitable transformation sequence is not found, the DTM will attempt to create a new transformation process. The DTM version 1 transformation creation process consists of the automated generation of XSLT documents to perform the transformation, although additional mechanisms are under development for DTM version 2. For example, if we have an XML input document that is valid with respect to one XML Schema, and a required XML output document that is valid with respect to a second XML Schema, the two formats can be represented as trees. In order to build the XSLT document that performs the transformation, the DTM first attempts to determine the relationships between the input and output XML elements, by considering the XML element names, or the XML element content formats. For example, if the input format has an element named *lat*, and the output format has an element named *latitude*, then the DTM may conclude that a relationship exists between the contents of these elements. However, the lat element may contain data as decimal degrees (e.g. 10.46), and the latitude element may contain data as degrees minutes and seconds (e.g. 10° 27' 36" N), so a conversion is required for the data contained in each lat elements. By searching a library of XSLT conversions (such as decimal degrees to degree, minutes, and seconds), and a database of element name relationships (such as *lat* to *latitude*), the DTM can build an entire XSLT document from library components. This XSLT document can then be applied to the immediate problem, and will also be saved and made available for future similar transformations. However, it is possible that the DTM will encounter an element for which it does not have a relationship. In such cases, the user is prompted to provide an indication of the correct relationship, which can then be used to create the current XSLT document. This new relationship will also be added to the DTM database, and will then be used if similar elements are encountered in the future. In this way, the ability of the DTM to automatically build XSLT documents is improved, and other users can benefit from the work of all previous users.

While XSLT documents are an effective method by which transformation rules can be described, they are not particularly fast when used to perform the transformations, especially for large datasets. To improve their efficiency, it is possible to instruct XSLT processors to automatically generate Java Byte Code (*.class* files) to perform the transformation. These *translets* typically perform the same transformation faster than the source XSLT documents, and as such can be used to increase throughput for the DTM. The current version of the DTM is implemented in Java, although it is possible that future versions will be ported to a higher performance language if testing warrants.

In order to interact with various data formats (such as XML documents, database recordsets, etc), and transformation methods (e.g. XSLT documents, executables, and scripts), the core DTM functionality is highly modular. As such, when new transformation methods are created in the future, the DTM functionality can easily be extended to include these new methods. For example, if a new XSLT document or executable is made available to the DTM it can be easily incorporated using the existing system. However, even if a fundamentally new processing method is developed, as is likely to occur as technology continues to advance, this new method can be used by IDACT by writing a new module, in a manner analogous to the approach used by device drivers to allow the Operating System and hardware devices to interact.

Programmatic access to the DTM is currently available though either a traditional Java based API, or through a Web Service based API that allows network enabled clients to

invoke the DTM functionality across the network. This approach ensures that local clients can make fast local connections via the Java API, while still allowing remote clients to access the functionality via the web service interface.

V. Future considerations

The development of the DTM will continue to incorporate a large number of predefined transformations, such as executables, scripts, and XSLT documents to perform common transformations for the Earth Science domain, and the wider NASA and scientific communities. Testing of the DTM is underway, with the first test environment being the University of Alaska SRP5 Sounding Rocket telemetry stream. Further test environments have been identified, and will be incorporated as the DTM development continues. The most significant task remaining for the DTM is the development and implementation of methodologies for the automated creation of new transformations. This will build on the results of the testing of the current implementation of the automated XSLT generation module, which itself will continue to be refined. The IDACT project will release the source code for all implemented tools under the GPL via *sourceforge.net*. An initial DTM toolkit will be released in July 2004, with future releases for the DTM and other IDACT components to follow as project milestones are reached.

Acknowledgments

This research is funded through the NASA Advanced Information Systems Technology Program (NASA award AIST-02-0135) and the University of Alaska Fairbanks.

References

[1] C. Crewdson and K. Nance, IDACT Data Source Registry Technical Specification, May 2004.

[2] S. Das,, K. Shuster and C. Wu, "ACQUIRE: agent-based complex query and information retrieval engine." *Proceedings of the first international joint conference on autonomous agents and multiagent systems*. Bologna, Italy, 2002.

[3] B. Hay and K. Nance, IDACT Transformation Manager Technical Specification. May 2004.

[4] B. Hay and K. Nance, "Simon: An Intelligent Agent For Heterogeneous Data Mapping." *Proceedings of the International Conference on Intelligent Systems and Control.* Honolulu, Hawaii. August 13-18, 2000.

[5] K. Nance, "Data System Planning For Formerly Used Defense Sites (FUDS)." *Proceedings of the American Society of Business and Behavioral Sciences: Government And Business Problems*. February 20-26, 1997.

[6] K. Nance, "Decision Support and Data Mining." *Proceedings of the International Simulation Multiconference*. April 6 – 10, 1997.

[7] K. Nance, "Synthesis of Heterogeneous Data Sets." *Proceedings of the 9th Annual Software Technology Conference*. May 6 – 10, 1997

[8] K. Nance, "Applying AI Techniques In Developing Plans For Formerly Used Defense Sites (FUDS) In Alaska." *Mathematical Modeling and Scientific Computing, vol. 8*, 1997.

[9] K. Nance and J. Wiens, "SynCon: Simulating Remediation Planning." *Mathematical Modeling and Scientific Computing,* 1998.

[10] K. Nance, J. Wiens and S. George, "The SynCon Project: Arctic Regional Environmental Contamination Assessment." *Proceedings of the38th Annual Western Regional Science Conference*. February, 1999.

[11] K. Nance, J. Wiens and S. George, "The SynCon Project: Phase II Assessing Human Health In the Arctic." *Proceedings of the International ICSC Congress on Computational Intelligence: Methods And Applications*. June, 1999.